United States Department of Commerce
National Oceanic and Atmospheric Administration

Project CRAFT Network Topology and Initial Analysis of Network Loads

National Severe Storms Laboratory Report (2002-01)

Prepared by: Yen Soon Low, University of Oklahoma, School of Computer Science
Dr. Lakshmivarahan, University of Oklahoma,
School of Computer Science
Kevin Kelleher, National Severe Storms Laboratory

Funding for this project provided by a grant from the NOAA High Performance Computing and Communications Office

January 18, 2002

National Severe Storms Laboratory
1313 Halley Circle
Norman, Oklahoma 73069

Table of Contents

Chapter 1 Introduction

The US National Weather Service (NWS) has recently completed the installation of 120 WSR-88D (Weather Surveillance Radar, 1988-Doppler) systems [Crum 93] as part of its $4.5B Modernization and Associated Restructuring Development Plan. Concurrently, there are another 26 Department of Defense and 12 Federal Aviation Administration radars being deployed, for a total of 158 radars of which 141 are in the continental United States.

The WSR-88D radars were originally designed without the archival capability, lacking proper devices for archiving large amount of data. As the scientific research value of the data was recognized, an interim solution was developed to provide the recording and long-term archival of the full-volume, full-precision Level II data (also known as base data). This is done based on 8 mm tape technology, which is extremely human-resource intensive, costly, inefficient, and unreliable. Currently, the archive for the 120 NWS radars is only approximately 65% complete.

In an attempt to begin addressing the long-term needs for WSR-88D base data transmission and archival, the Center for Analysis and Prediction of Storms (CAPS) at the University of Oklahoma joined forces in 1998 with the University Corporation for Atmospheric Research, the University of Washington, the National Severe Storms Laboratory, and the WSR-88D Operational Support Facility to establish the Collaborative Radar Acquisition Field Test (CRAFT). The principal goal of CRAFT is to demonstrate the real time compression and internet-based transmission of WSR-88D base data from multiple radars with a view toward nationwide implementation.

In this report, an overview of the project CRAFT is given. In particular, the purpose of collecting base data and the system configuration used to transmit the data are being discussed. Then, the underlying network structure is examined in order to understand the network properties and constraints. These parameters play an important role in providing the knowledge for further improvements of the network. A simulation using Network Simulator (NS) is done to aid the analysis.

Chapter 2 CRAFT Data and System Configuration

Raw data collected by the WSR-88D radar are used to generate four Archive Level data sets [OFCM 91]. The Archive Level II data (base data) are full precision, full spatial resolution data of the three Doppler moments (radial wind, reflectivity, and spectrum width).

The value of base data in research is tremendous as noted in [Droegemeier 00]. For example, it is valuable in the following research areas:

1. Storm-scale numerical weather prediction
2. Storm-scale variability
3. Synthetic storm feature climatologies

Since the base data play a very important role in the research community, it is essential that the data can be accessed (or archived) in a convenient and efficient manner.

2.1 Legacy 8 mm Tape Archival Process

The archival of base data was complicated by the large volume of information needed to be stored. Figure 2.1 below shows a schematic of the base data archival process before the CRAFT project's introduction of real-time delivery of Level II data. The new archival process for real-time data is discussed in Section 2.2.

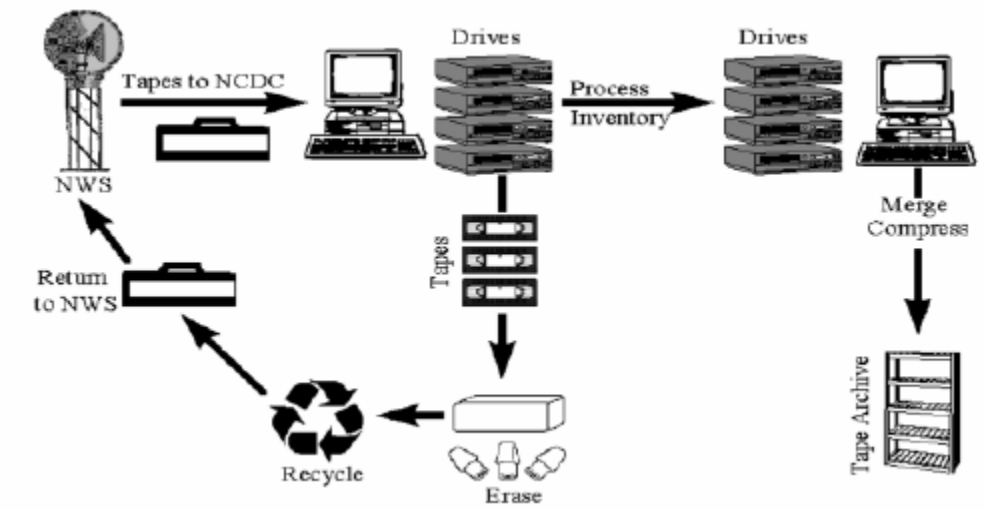


Figure 2.1 NCDC pre-CRAFT archival procedure for WSR-88D Level II data [Droegemeier 00]

The process begins when a radar site mails a set of ten 8 mm tapes to the NCDC. Each tape may contain up to 4.8 gigabytes of data. The data are then merged and compressed, at a rate up to 10:1, to another archive tape (IBM 3590) for long-term storage. The original 8 mm tapes are then recycled.

On average, the National Climatic Data Center (NCDC) receives 165 gigabytes of base data each day, or approximately 60 terabytes annually. By late 1999, the NCDC had archived some 42,000 8 mm tapes. The base data archive for all 120 NWS radars was 64.9% complete, in comparison to 35.7% and 9.5% for the Department of Defense and Federal Aviation Administration radars, respectively.

The slow speed of the 8 mm tape drives significantly lengthens the time needed to access the base data. Therefore, an order for even a modestly large data set involving a dozen radars may take a few months to be processed and the cost for the data can run into hundreds of thousands of dollars. Furthermore, no mechanism presently exists to peruse the base data and the cost to maintain the now outdated tape system exceeds $0.75M per year. As a result, the low usage of base data does not reflect a lack of interest, but rather a lack of cost-effective access.

2.2 CRAFT System Configuration

There are *two* major components that enable the real-time delivery of the base data. The first is the *Radar Interface and Data Distribution System* (RIDDS) [Jain 95], which was created by the NSSL to extract base data from the NEXRAD radar product generator in real time. RIDDS consists of a Sun SparcStation 5/110 workstation, running special software developed by the NSSL that takes base data from the wideband-3 port or the RPG and outputs them as UDP packets to an Ethernet hub. A circular buffer on the RIDDS is used to hold several volume scans of data before it is transmitted.

The second component is the *Local Data Manager* (LDM) software developed by the UCAR Unidata program. LDM is a system for event-driven data distribution that acquires data from, and transmits data to other networked computers.

As shown in Figure 2.2, CRAFT involves connecting a RIDDS workstation to the radar, along with a personal computer running the Linux operating system and LDM software. The inexpensive PC requires only nominal memory (64 Mbytes) and disk space (2 Gigabytes). The data are compressed by the PC, and then sent to an optional multi-port router ($1,500) and finally to the Internet via a direct Internet connection, a T1 line, or a

56kb line. The data are then passed to a large LDM server on the Abilene backbone for delivery to multiple users through Internet.

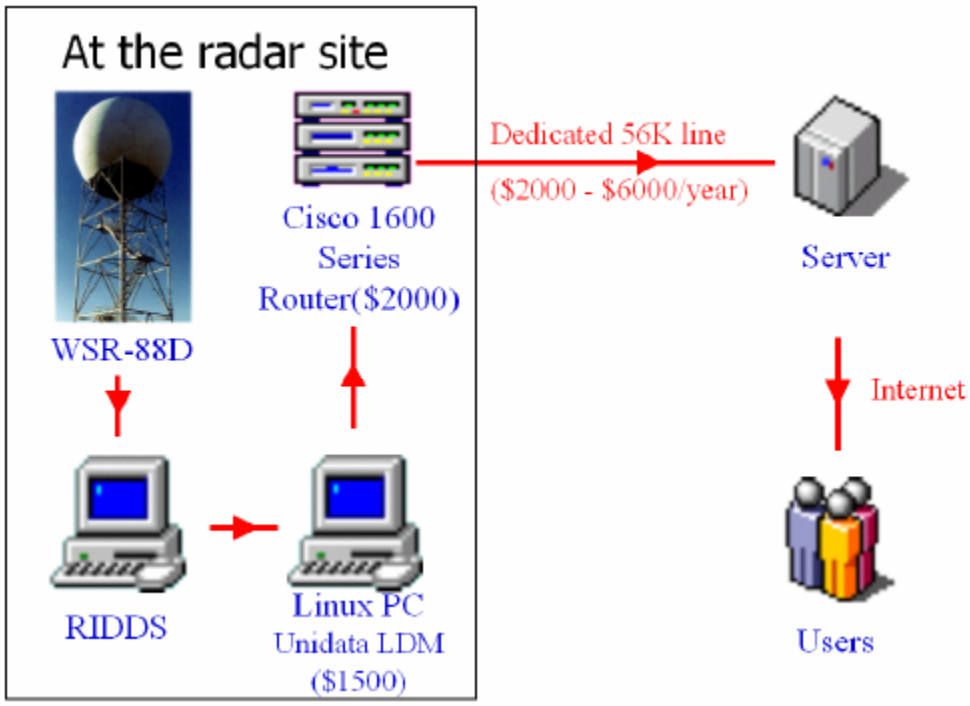


Figure 2.2 Data compression and transmission in CRAFT [Droegemeier 00]

In addition to the two major components described above, a key aspect of CRAFT is the used of a data compression algorithm known as BZIP2. This algorithm is an off-the-shelf, public domain, lossless data compression algorithm. It compresses the base data, in packets of 100 radials, down to an average of 1/10th their original size. This translates to a data volume of between 40 and 90 gigabytes/day for the 120 NWS radars, compared to 165 gigabytes without any data compression. The aggregate compressed base data rate for the entire national WSR-88D network is only 6-8 mbps.

Chapter 3 Network Structures

The archival and acquisition of base data involves the use of two national networking infrastructures: Internet2 and Abilene. These enabling network infrastructures are discussed first before the overall CRAFT network structure is presented.

3.1 Overview of Internet2 and Abilene Network

Internet2 is a non-profit consortium being led by over 180 universities working in partnership with industry and government to develop and deploy advanced network applications and technologies, accelerating the creation of tomorrow's Internet.

Abilene is an advance backbone network that supports the development and deployment of the new applications being developed within the Internet2 community. Abilene connects regional network aggregation point, which is called gigaPoPs (gigabits/second Point of Presence). Both Internet2 and Abilene are managed by the non-profit University Corporation for Advanced Internet Development (UCAID). The following shows the core Abilene network topology:

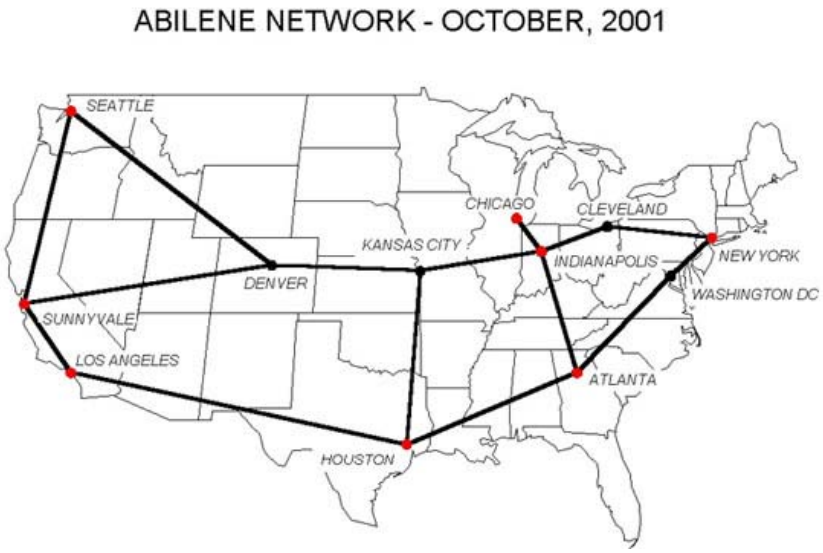


Figure 3.1 Abilene Core Network
(http://www.internet2.edu/abilene/html/maps.html)

The Abilene architecture is best viewed as having two components:

1. A Core Architecture, consisting of a set of router nodes connected to each other with interior circuits
2. An Access Architecture, consisting of a set of access circuits, each of which connects a router node to an Abilene connector

3.2 CRAFT Network

The archival and acquisition of base data is through a hierarchical network structure. Local radar is connected to the nearest Internet2 node via a T1 line or 56k phone line or in some cases, through the regional network.

By early 1999, six WSR-88D radars in and around Oklahoma were delivering base data in real time using the methodology given in Section 2.2. The topology of those radars is given below:

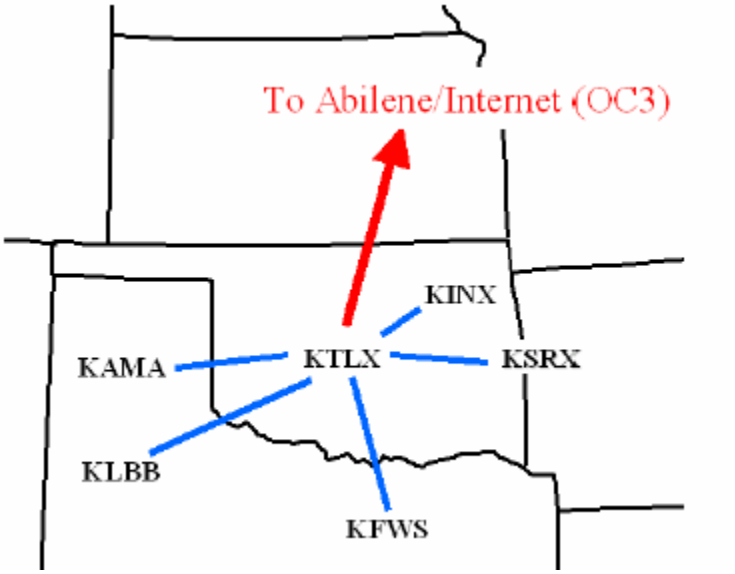


Figure 3.2 The original CRAFT network as of January, 1999 [Droegemeier 00]

In the early June 2000, the NCDC began receiving compressed base data in real time from the original six CRAFT radars via the commodity internet. These data were being transferred directly, and automatically, to the NCDC mass storage system as shown in the following:

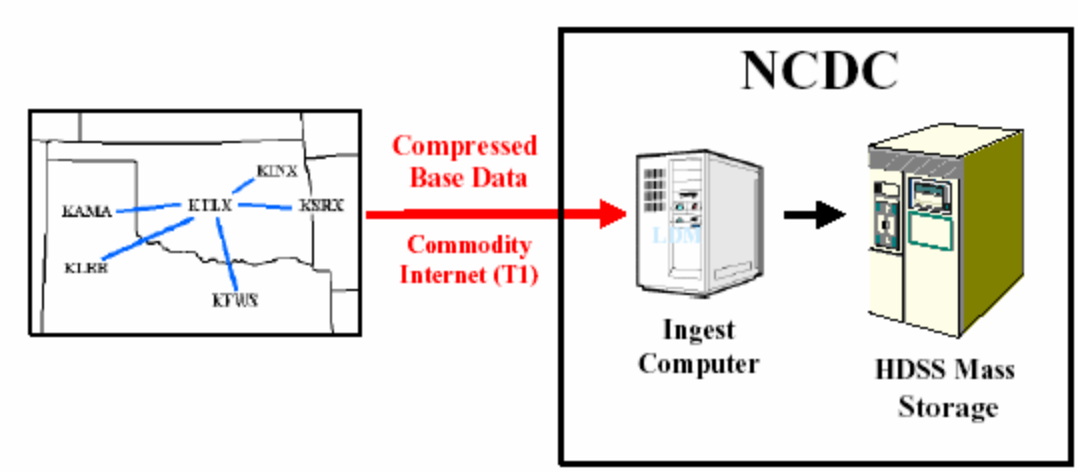


Figure 3.3 The real time archival process at the NCDC [Droegemeier 00]

The base data from CRAFT have been verified to be identical to those from the 8 mm tapes. The CRAFT system has reduced the number of archival processing steps from 8 to 2 and has reduced the time required to archive a data set by a factor of 4. Given the success of initial test bed of six radars in Oklahoma, other radars throughout the country have been added. The following figure shows the current topology of 21 CRAFT radars transmitting base data in real time as well as 18 new radars to be added soon in the future:

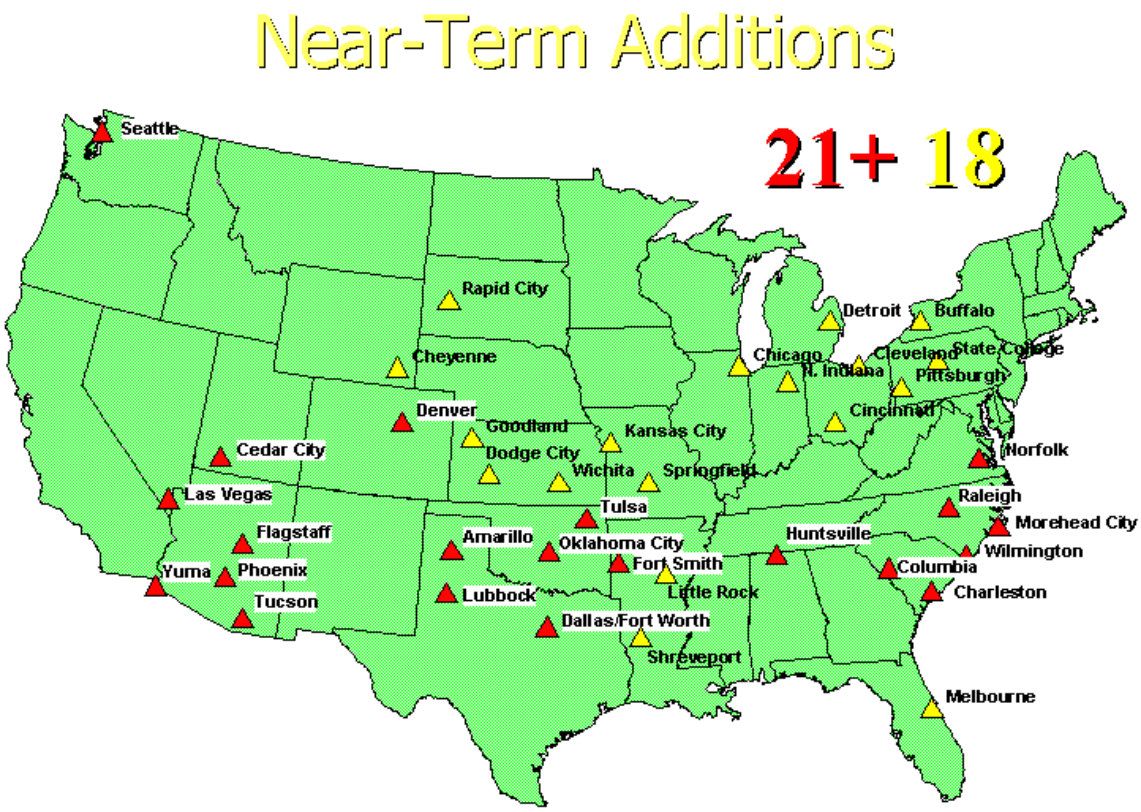


Figure 3.4 The CRAFT network as of October 2001 (http://kkd.ou.edu/new_craft_radars.htm)

Chapter 4 Network Simulation

In order to understand how well the CRAFT network performs and to predict its performance as a possible nationwide operational radar data delivery system in the future, it is crucial to examine the network in detail. In particular, network properties such as bandwidth, local buffer, and latency play a very important role in understanding the network performance. These parameters are studied through a simulation of the initial test bed of 6 radars in Oklahoma.

4.1 Simulation Using Network Simulator (NS)

The simulation is done using the Network Simulator (NS). NS (version 2) is an object-oriented, discrete event driven network simulator developed at UC Berkeley written in C++ and OTcl. NS is primarily useful for simulating local and wide area networks. It implements *network protocols* such as TCP and UDP, *traffic source behavior* such as FTP, Telnet, Web, CBR and VBR, *router queue management mechanism* such as Drop Tail, RED and CBQ, and *routing algorithms* such as Dijkstra, and more. Due to its implementation of a wide variety of network functionalities, NS was determined to be a suitable candidate for simulating the CRAFT network.

There are two languages, namely C++ and OTcl, that comprise the NS simulation software package. C++, which can efficiently manipulate bytes, packet headers, and implement algorithms that run over large data sets, is suitable for detailed protocol implementation. On the other hand, OTcl enables fast exploration of a number of scenarios due to its nature of simple syntax. The CRAFT network simulation was written in OTcl only, making use of the default NS built-in classes (Appendix A).

The NS outputs several text files containing statistics information (Appendix B) and information for the input to a graphical display tool called Network Animator (NAM). The following figure shows a snapshot of the graphical representation of the CRAFT network in NAM:

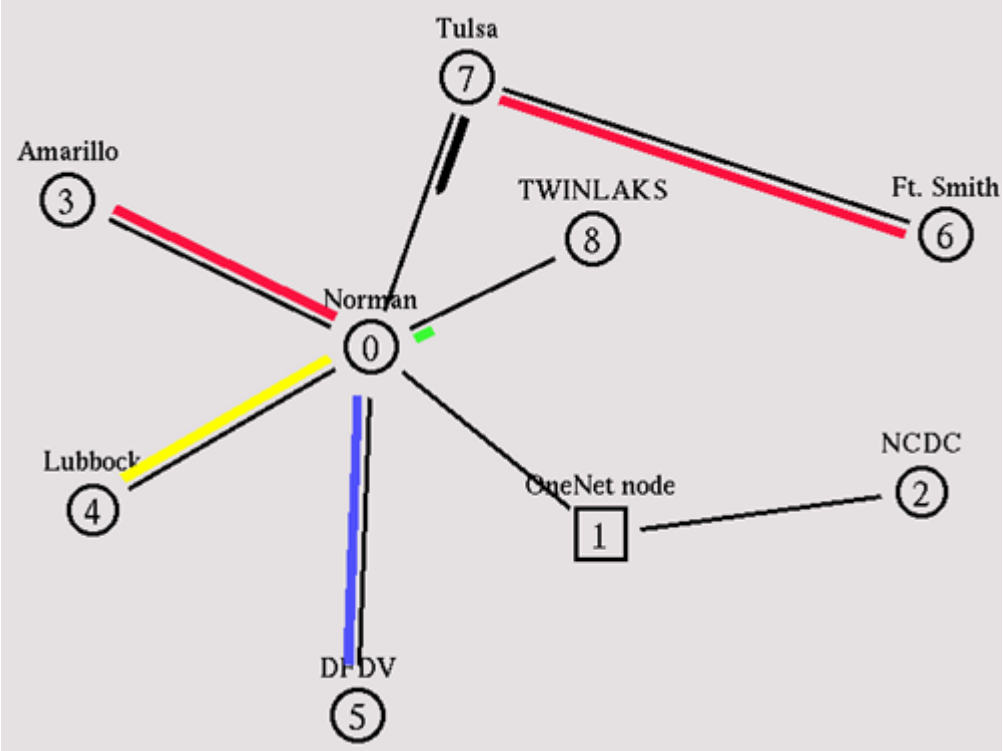


Figure 4.1 A snapshot of CRAFT network in NAM

4.2 Quantitative Analysis

There are 3 major aspects in the analysis for the CRAFT network: bandwidth, local buffer (queue), and latency. Bandwidth examines whether the link usage is close to optimal, which then results in being queued in the local buffer. Local buffer examines whether the buffer size is adequate for storing several days' worth of data before data loss happens. Latency examines the delay in the packet transfer, which is an important key to deciding the usability of data in a time-critical event.

4.2.1 Bandwidth

When the radar generates data in a rate of which it exceeds the link capacity, the data are stored temporary in a queue in LDM. The data generation rate is based heavily on the weather condition. On average, the compressed data rate is approximately 40 kbps. In a heavy weather condition, the rate might exceed 56 kbps.

The following figure shows the capacity of each link in the CRAFT network in Oklahoma:

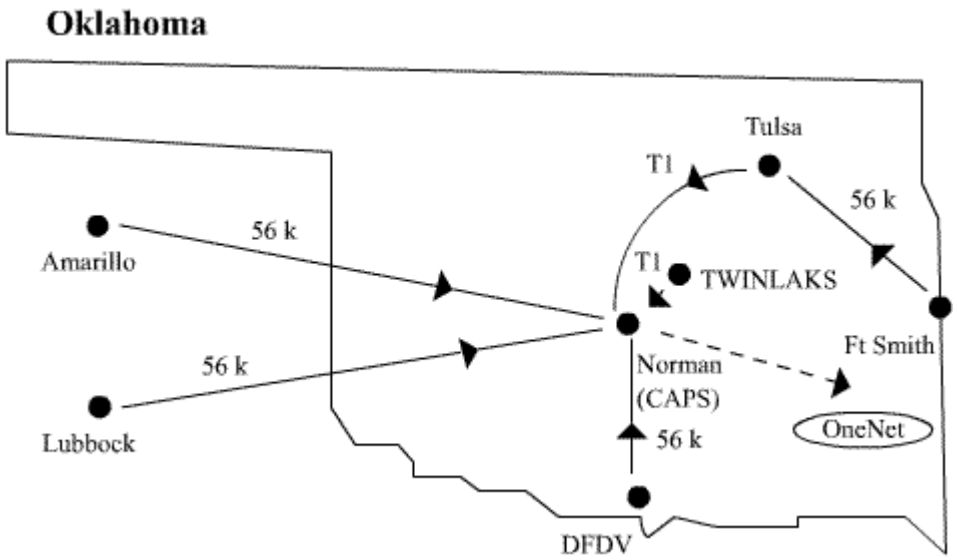


Figure 4.2 CRAFT network topology in Oklahoma

OneNet is connected to Abilene using a T3 line (45 Mbps). The Abilene backbone is connected with a speed of 2.4 gigabit/seconds. Therefore, bandwidth is not an issue when considering the aggregated data transfer of 6 radars producing only several hundreds of kilobits per second from Norman to NCDC (through OneNet and Abilene).

On the other hand, some radars are connected to the Internet using 56 kbps phone lines. This causes problem when the data are generated at a rate higher than 56 kbps. Therefore, for this particular situation, the local buffer is needed to store the data (either in memory or on disk) in order to prevent data loss.

4.2.2 Local Buffer

It is interesting to know when the buffer will overflow during a heavy weather condition in which data are generated at a very high rate. Given the buffer size of 100 MB, Figure 4.3 below shows the time duration at which the buffer overflows given different data generation rates:

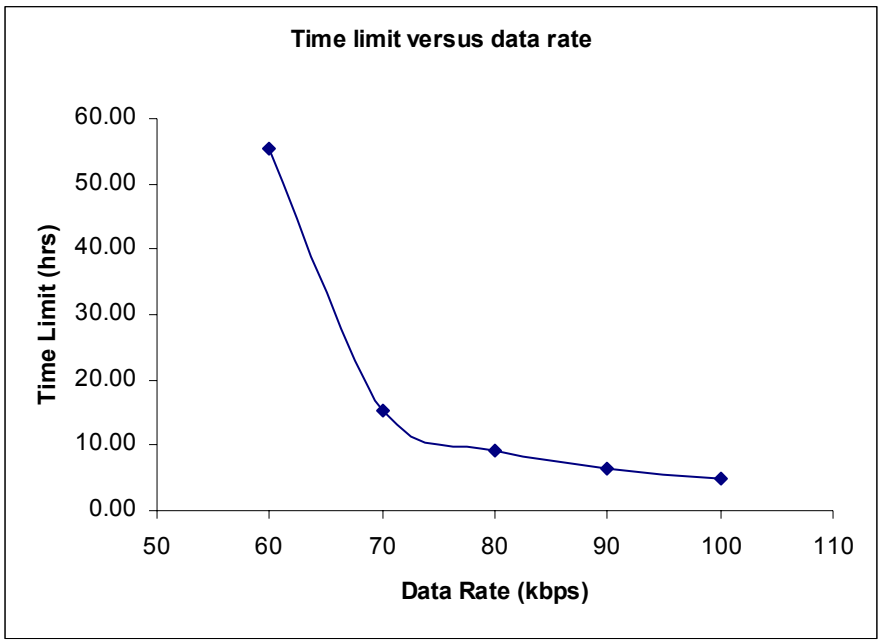


Figure 4.3 Duration before buffer overflows for different data rates (Buffer = 100 MB)

From the figure above, it can be seen that the buffer can withstand approximately 25 hours before overflows for a data rate of 65 kbps. Since the data generation rate is fluctuating in real situation, the time cannot be precisely predicted. The figure above shows that a buffer size of 100 MB is adequate if the data rate does not exceed 70 kbps for a long period of time. Given that the data rate seldom exceeds 70 kbps, the current buffer is large enough for its purpose.

4.2.3 Latency

The latency between two nodes can be calculated as follows:

Latency = Transmit Time + Link Propagation + Queue Delay

Transmit Time = Size/Bandwidth

Link Propagation = Distance/Speed of Light

Since the transmit time and link propagation is fixed for a given topology and the total amount of data being sent, latency is mainly affected by the queuing delay. For the CRAFT network, the queuing delay includes the delay at the local buffer and the delay in routers during the transfer through internet.

The delay in routers for the internet is highly dynamic, thus it is very difficult to predict accurately the delay. On the other hand, the delay of data in the local buffer is strictly the transfer rate of data from PC onto the link. Therefore, the local delay can be predicted quite accurately. The following shows a plot of average queue delay for different durations of simulation time at a data transmission rate of 100 kbps:

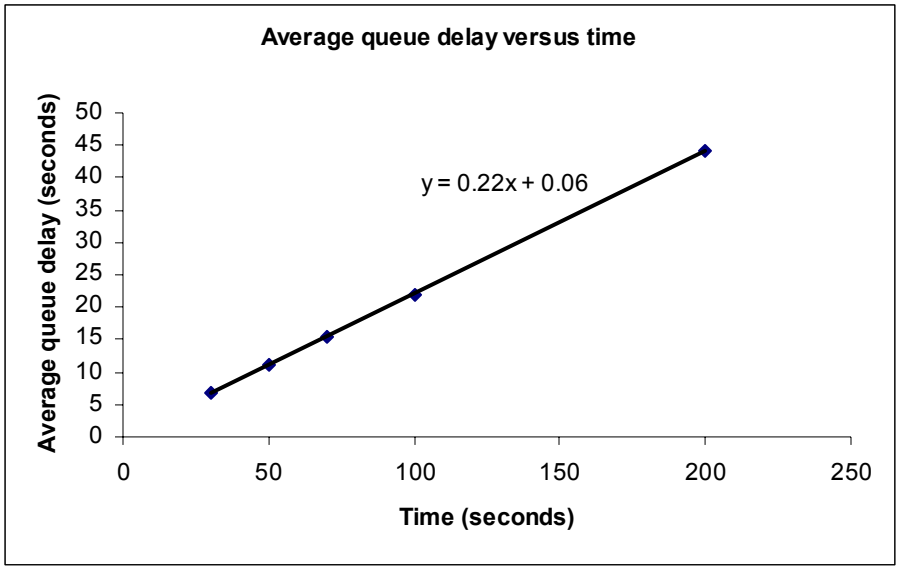


Figure 4.5 Average queue delay versus simulation time (transmission rate = 100 kbps)

From the figure above, for example, it can be seen that for data rates of 100 kbps for 6 hours, the average queue delay is calculated to be 1.3 hours. At the time of this writing, the actual and simulated latency is still being investigated to see how well the numbers correlate with each other.

4.3 Scalability

The CRAFT network is hierarchical; the local radar is connected to local PC, all the regional PCs are connected to the high-speed Abilene backbone. This structure is highly scalable because any number of local PCs can simply be added to the network without causing any delay due to the high speed of the backbone. This is an extremely favorable feature because project CRAFT aims to design a network to incorporate all 120 nationwide radars in the future.

Chapter 5 Conclusions and Future Work

A principal benefit of CRAFT, particularly in terms of cost and reliability, is its utilization of existing software (e.g., LDM, BZIP2, Linux) and networks (Abilene and the commodity Internet). This framework lays the foundation for the cost-effective expansion of CRAFT to all WSR-88D radars by the NWS.

Although the Internet itself might be viewed as unreliable for use in time-sensitive meteorological operations, the CRAFT results have proven otherwise. Indeed, the Internet is so pervasive and redundant that it probably represents one of the most reliable means of data transmission currently available.

In conclusion, project CRAFT allows the archival of the radar base data easily and reliably and it facilitates real-time access through the Internet in a very cost-effective and efficient manner.

Project CRAFT is an ongoing project, therefore it is still in its experimental phase. The following outlines the future work to be done for further enhancement and improvement on the project:

1. One principal limitation of Project CRAFT is the single point of failure in the communications link from the radar to the Internet. Therefore, some redundancies may be needed in the future to provide a certain degree of reliability.
2. In the new Open Systems Architecture, the functionality of the RIDDS and LDM computers will be replaced by a more powerful single machine known as the Base Data Distribution System (BDDS).
3. Ability to allow data access in real time to all appropriate users by minimizing delay through hardware and software improvements.

References

[Crum 93] Crum, T.D., R.L. Alberty, and D.W. Burgess, 1993: Recording, archiving and using WSR-88D data. *Bull. Amer. Meteor. Soc.*, **74**, 645-653

[Droegemeier 00] Droegemeier, K.K., K. Kelleher, T. Crum, J.J. Levit, et. al. Project CRAFT: A Test Bed for Demonstrating the Real Time Acquisition and Archival of WSR-88D Base (Level II) Data. *Draft for Bull. Amer. Meteor. Soc.*, 1 October 2000

[Jain 95] Jain, M. and D. Rhue, WSR-88D's live data stream now accessible in real-time. *NSSL Briefings*, summer 1995, 11

[OCFM 91] Doppler radar meteorological observations, Part A. System concepts, responsibilities, and procedures. Federal Meteorological Handbook No. 11, FCM-H11A-1991 (Interim Version One), U.S. Government Printing Office, Washing, DC, 58pp

CRAFT website - http://kkd.ou.edu/craft.htm

Internet2 website - http://www.internet2.edu/

Abilene website - http://www.internet2.edu/abilene/

OneNet website - http://www.onenet.net/

Network Simulator website - http://www.isi.edu/nsnam/ns/

Network Simulator Tutorial - http://nile.wpi.edu/NS/,
http://www.isi.edu/nsnam/ns/tutorial/index.html

Appendix A

NS Simulation Source Code

```
#Create a simulator object
set ns [new Simulator]

#Open the nam trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Open files for queue and statistics information
set qf [open q.tr w]
set qf1 [open q1.tr w]
set sf [open stat.tr w]

#Define color index
$ns color 0 red
$ns color 1 yellow
$ns color 2 blue
$ns color 3 green
$ns color 4 black
$ns color 5 chocolate
$ns color 6 brown
$ns color 7 tan
$ns color 8 gold

#Define packet information
set myPacketSize 1000   ;# in bytes
set myRate 100          ;# in kilobits

#Define queue information
set queuelimit 100      ;# in MBytes
set queuesize [expr $queuelimit * 1000000 / $myPacketSize]

#Running time
set interval 30

#Define a 'finish' procedure
proc finish {} {
        global ns nf qf sf qf1
        $ns flush-trace

        close $nf
        close $qf
        close $qf1
        close $sf

        exec nam out.nam &
        exit 0
}

#Write information into the stat file
proc write-stats-info {node myqmon mysamples} {
        global sf interval

        puts $sf "Node: $node"
        puts $sf "---------------packet-----------------"
        puts $sf "# of packet arrivals = [expr [$myqmon set parrivals_]/$interval]/seconds"
        puts $sf "# of packet departures = [expr [$myqmon set pdepartures_]/$interval]/seconds"
        puts $sf "# of packet drops = [$myqmon set pdrops_]"
```

```
        puts $sf "# of packets in queue when ended = [$myqmon set pkts_]"
        set pktint [$myqmon get-pkts-integrator]
        puts $sf "Integral queue size = [$pktint set sum_]"

        puts $sf "---------------bytes-------------------"
        puts $sf "# of bytes arrival = [expr [$myqmon set barrivals_]/$interval]/seconds"
        puts $sf "# of bytes departure = [expr [$myqmon set bdepartures_]/$interval]/seconds"
        puts $sf "# of bytes drop = [$myqmon set bdrops_]"
        puts $sf "# of bytes in queue when ended = [$myqmon set size_]"
        set byteint [$myqmon get-bytes-integrator]
        puts $sf "Integral queue size = [$byteint set sum_]"
        puts $sf ""
        puts $sf "Average queue delay = [$mysamples mean]"
        puts $sf ""
        puts $sf ""
}

proc attach-cbr-traffic {node mysink size rate mycolor} {
        #Get an instance of the simulator
        set ns [Simulator instance]

        #Create a TCP agent and attach it to the node
        set source [new Agent/UDP]
        #$source set packetSize_ $size
        $ns attach-agent $node $source
        $source set class_ $mycolor

        #Create a constant bit rate traffic and set its configuration parameters
        set traffic [new Application/Traffic/CBR]
        $traffic set packetSize_ $size
        $traffic set rate_ [append rate "kb"]

        #Attach traffic source to the traffic generator
        $traffic attach-agent $source

        #Connect the source and the sink
        $ns connect $source $mysink
        return $traffic
}

#Create nodes
set norman [$ns node]
set onenet [$ns node]
set ncdc [$ns node]
set amarillo [$ns node]
set lubbock [$ns node]
set dfdv [$ns node]
set ftsmith [$ns node]
set tulsa [$ns node]
set twinlaks [$ns node]

#Label node
$norman label "Norman"
$onenet label "OneNet node"
$ncdc label "NCDC"
$amarillo label "Amarillo"
$lubbock label "Lubbock"
```

```
$dfdv label "DFDV"
$ftsmith label "Ft. Smith"
$tulsa label "Tulsa"
$twinlaks label "TWINLAKS"

#Define special characteristics
$onenet shape "box"

#Create duplex-link
$ns duplex-link $norman $onenet 45Mb 10ms DropTail
$ns duplex-link $onenet $ncdc 2400Mb 10ms DropTail
$ns duplex-link $amarillo $norman 56kb 10ms DropTail
$ns duplex-link $lubbock $norman 56kb 10ms DropTail
$ns duplex-link $dfdv $norman 56kb 10ms DropTail
$ns duplex-link $ftsmith $tulsa 56kb 10ms DropTail
$ns duplex-link $tulsa $norman 1.54Mb 10ms DropTail
$ns duplex-link $twinlaks $norman 1.54Mb 10ms DropTail

#$ns duplex-link-op $amarillo $norman queuePos 0.5

#Set the queue limit
$ns queue-limit $norman $onenet $queuesize
$ns queue-limit $amarillo $norman $queuesize
$ns queue-limit $lubbock $norman $queuesize
$ns queue-limit $dfdv $norman $queuesize
$ns queue-limit $ftsmith $tulsa $queuesize
$ns queue-limit $tulsa $norman $queuesize
$ns queue-limit $twinlaks $norman $queuesize

#Create queue monitor for the norman station
set qmon [$ns monitor-queue $norman $onenet $qf]
set samples [new Samples]
$qmon set-delay-samples $samples

#Create queue monitor for the amarillo station
set qmon1 [$ns monitor-queue $amarillo $norman $qf1]
set samples1 [new Samples]
$qmon1 set-delay-samples $samples1

for {set i 0} {$i<6} {incr i} {
        set sink($i) [new Agent/Null]
        $ns attach-agent $norman $sink($i)
}

set sink_tulsa [new Agent/Null]
$ns attach-agent $tulsa $sink_tulsa

set sink_ncdc [new Agent/Null]
$ns attach-agent $ncdc $sink_ncdc

set doubleRate [expr $myRate * 2]
set totalRate [expr $myRate * 6]

#Create traffic sources
set cbr0 [attach-cbr-traffic $amarillo $sink(0) $myPacketSize $myRate 0]
set cbr1 [attach-cbr-traffic $lubbock $sink(1) $myPacketSize $myRate 1]
set cbr2 [attach-cbr-traffic $dfdv $sink(2) $myPacketSize $myRate 2]
```

```
set cbr3 [attach-cbr-traffic $twinlaks $sink(3) $myPacketSize $myRate 3]
set cbr4 [attach-cbr-traffic $tulsa $sink(4) $myPacketSize $doubleRate 4]
set cbr5 [attach-cbr-traffic $ftsmith $sink_tulsa $myPacketSize $myRate 0]
set cbr6 [attach-cbr-traffic $norman $sink_ncdc $myPacketSize $totalRate 2]

puts $sf "**************************"
puts $sf "Durations = $interval seconds"
puts $sf "Date Rate = $myRate kbits"
puts $sf "Packet Size = $myPacketSize bytes"
puts $sf "**************************"
puts $sf ""

$ns at 0.11 "$cbr0 start"
$ns at 0.12 "$cbr1 start"
$ns at 0.13 "$cbr2 start"
$ns at 0.14 "$cbr3 start"
$ns at 0.15 "$cbr4 start"
$ns at 0.16 "$cbr5 start"
$ns at 0.17 "$cbr6 start"

$ns at [expr $interval + 0.11] "$cbr0 stop"
$ns at [expr $interval + 0.12] "$cbr1 stop"
$ns at [expr $interval + 0.13] "$cbr2 stop"
$ns at [expr $interval + 0.14] "$cbr3 stop"
$ns at [expr $interval + 0.15] "$cbr4 stop"
$ns at [expr $interval + 0.16] "$cbr5 stop"
$ns at [expr $interval + 0.17] "$cbr6 stop"

$ns at [expr $interval + 0.4] "write-stats-info norman $qmon $samples"
$ns at [expr $interval + 0.4] "write-stats-info amarillo $qmon1 $samples1"

$ns at [expr $interval + 0.7] "finish"

#Run the simulation
$ns run
```

Appendix B

Sample Output Statistics File

****
Durations = 30 seconds
Date Rate = 100 kbits
Packet Size = 1000 bytes
****

Node: norman
----------packet----------
# of packet arrivals = 75/seconds
# of packet departures = 75/seconds
# of packet drops = 0
# of packets in queue when ended = 0
Integral queue size = 0
----------bytes----------
# of bytes arrival = 75033/seconds
# of bytes departure = 75033/seconds
# of bytes drop = 0
# of bytes in queue when ended = 0
Integral queue size = 0

Average queue delay = 0

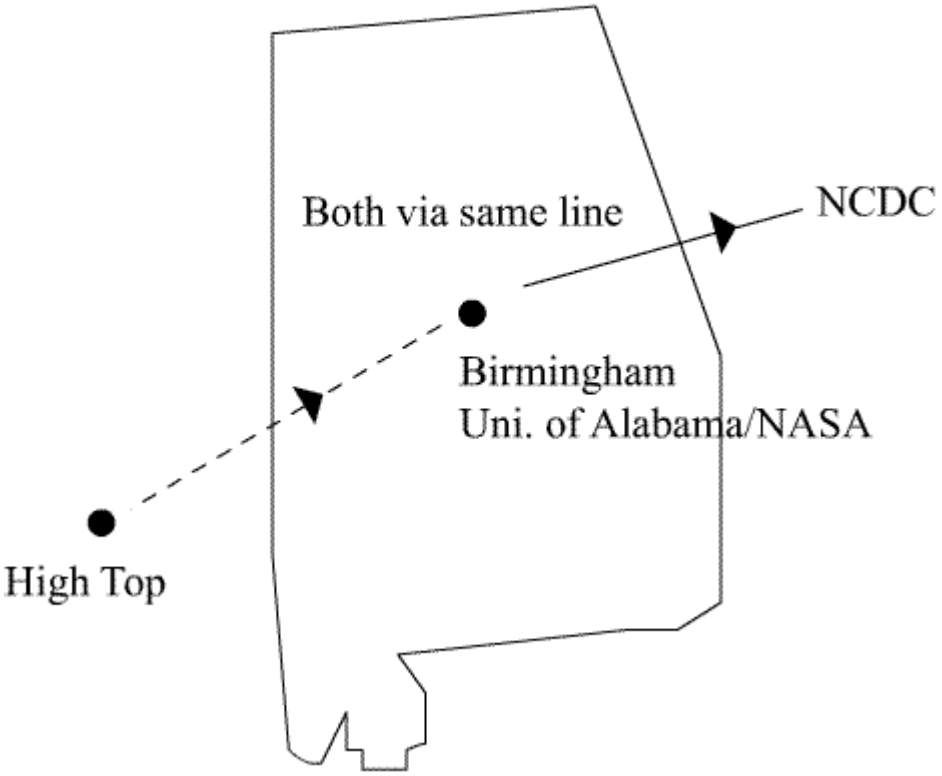
Node: amarillo
----------packet----------
# of packet arrivals = 12/seconds
# of packet departures = 7/seconds
# of packet drops = 0
# of packets in queue when ended = 163
Integral queue size = 2522.0000000000073
----------bytes----------
# of bytes arrival = 12533/seconds
# of bytes departure = 7100/seconds
# of bytes drop = 0
# of bytes in queue when ended = 163000
Integral queue size = 2522000.0000000042

Average queue delay = 6.66286

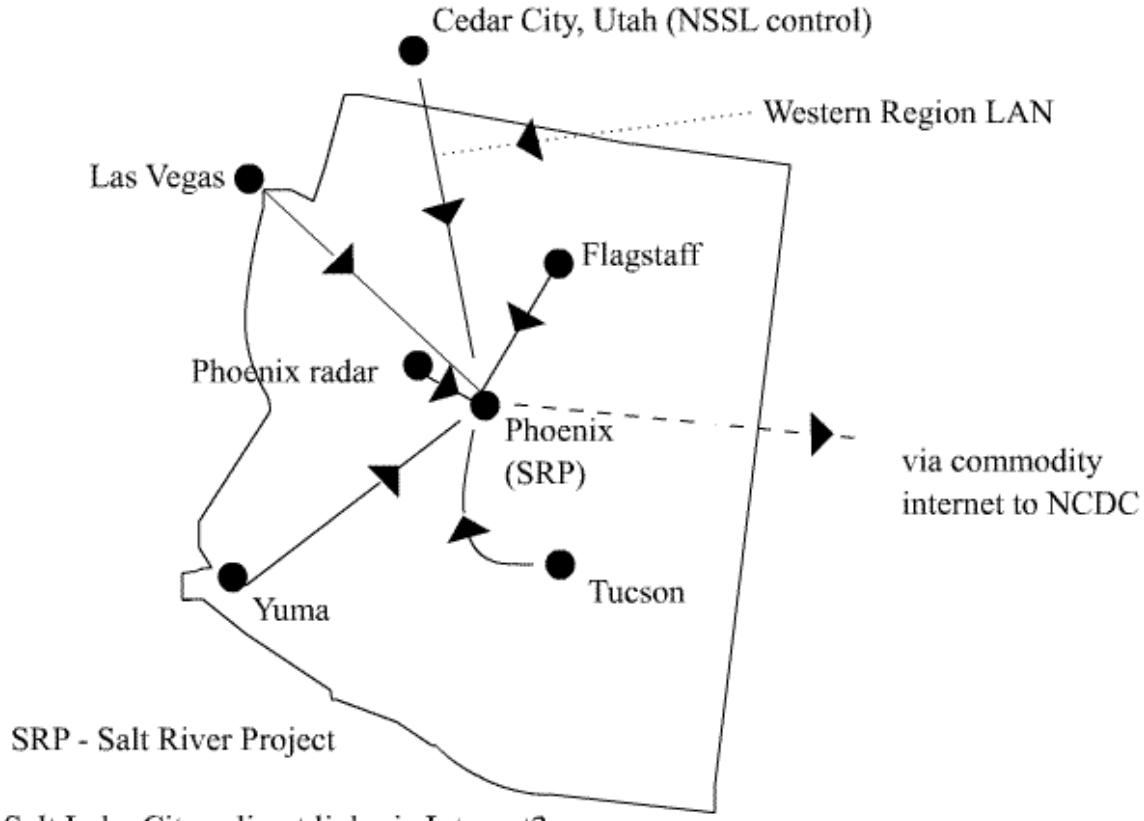
Appendix C

Different State Topologies for Project CRAFT

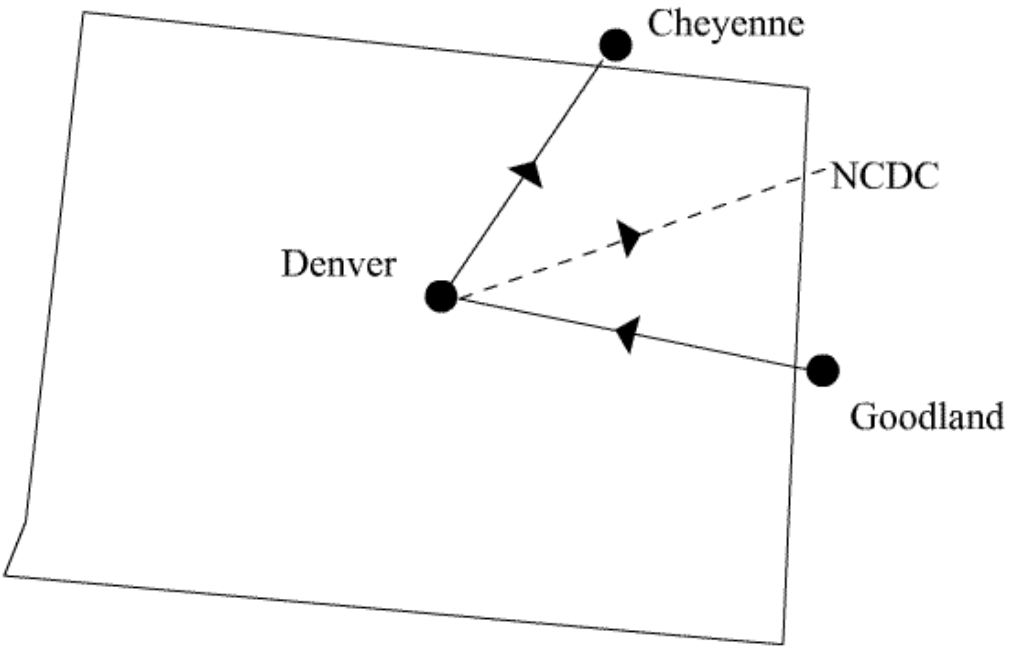
Alabama


Both via same line
NCDC
Birmingham
Uni. of Alabama/NASA
High Top

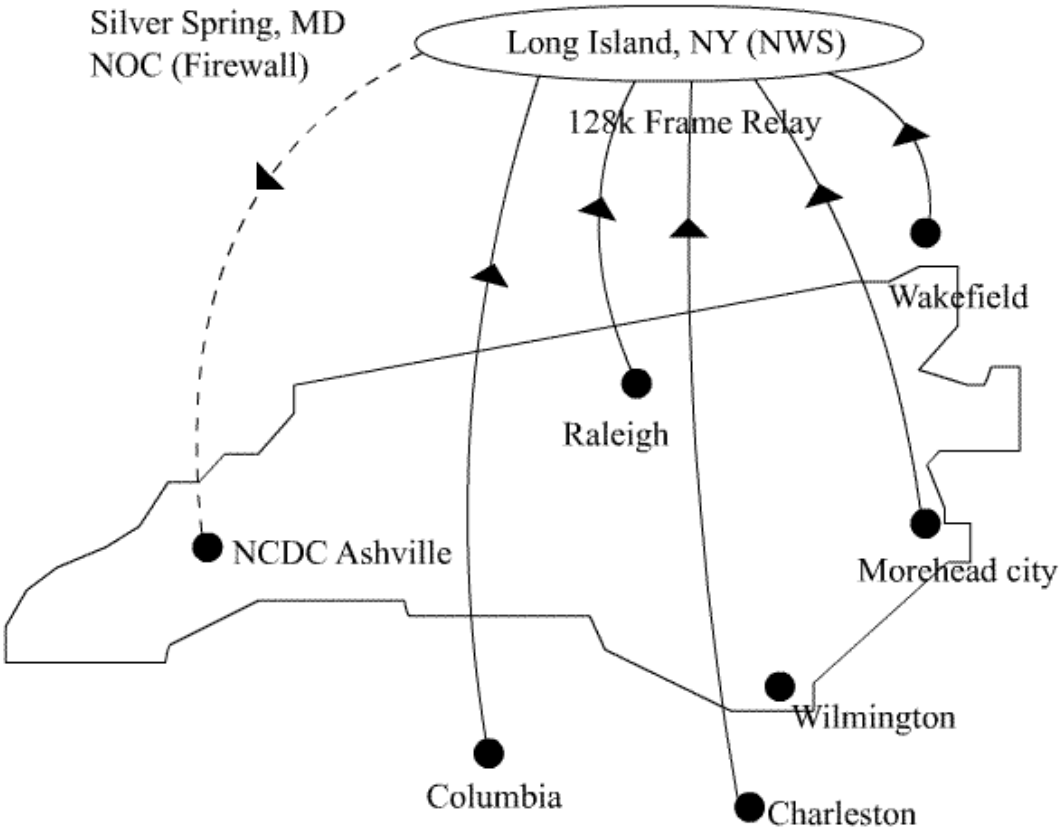
Arizona


Cedar City, Utah (NSSL control)
Western Region LAN
Las Vegas
Flagstaff
Phoenix radar
Phoenix
(SRP)
via commodity
internet to NCDC
Tucson
Yuma
SRP - Salt River Project
Salt Lake City - direct link via Internet2

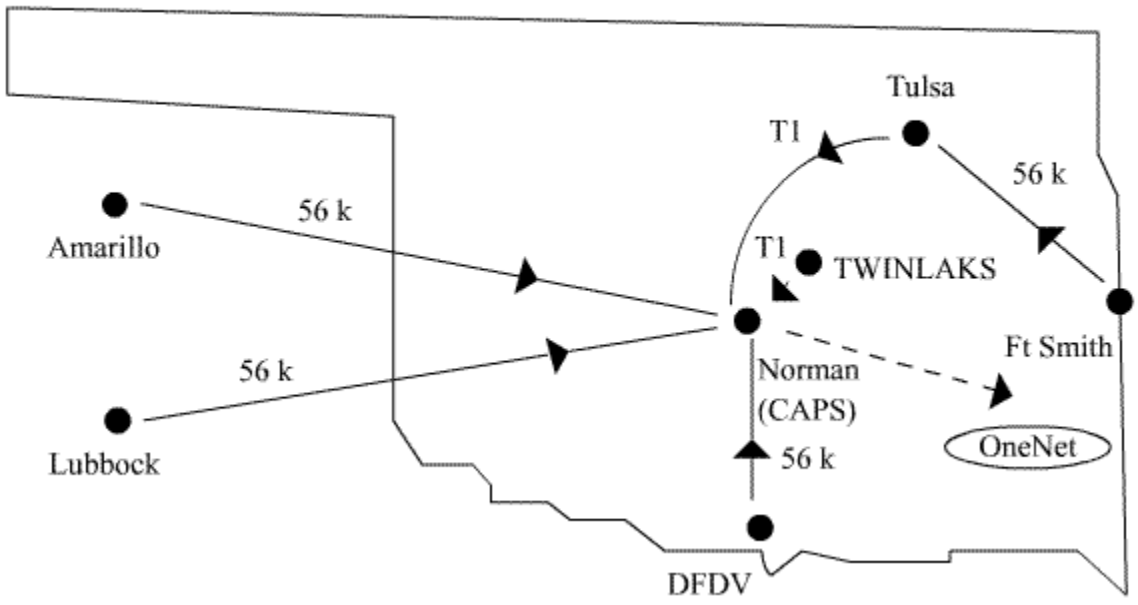
Colorado



North Carolina



Oklahoma


Tulsa
T1
56 k
Amarillo
56 k
T1
TWINLAKS
Ft Smith
56 k
Norman
(CAPS)
Lubbock
OneNet
56 k
DFDV